

## AI-POWERED EMOTION DETECTION FOR VIRTUAL HEALTHCARE CONSULTATIONS

**A dissertation submitted in partial fulfillment of the requirements for the award of the Degree of**

# Bachelor of Technology

**In**

# Computer Science and Engineering

**By**

# MD SAMREEN BEGUM

## (23U61A0544)

### Under the guidance of Mrs. G Pavani

B. Tech., M.Tech.,Phd.

### Assistant Professor

****

**A NAAC Accredited Institution**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## GLOBAL INSTITUTE OF ENGINEERING & TECHNOLOGY

**(Approved by AICTE, New Delhi & Affiliated to JNTUH) (Recognized under Section 2(f) of UGC Act 1956)**

**An ISO:9001-2015 Certified Institution CHILKUR (V), MOINABAD (M), R.R. DIST. T.S - 501504**

**MAY-2025**



**(Approved by AICTE & Affiliated to JNTUH) (Recognized under Section 2(f) of UGC Act 1956) An ISO:9001-2015 Certified Institution**

**Survey No. 179, Chilkur (V), Moinabad (M), Ranga Reddy Dist. TS.**

**JNTUH Code (U6) CIVIL – CSE – CSM – CSD - MECH – ECE – MBA – M.Tech. EAMCET Code– GLOB**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**Mrs. Noore Ilahi Date :27/05/2025**

B. Tech., M. Tech.

**Assistant Professor & Head**

# CERTIFICATE

This is to certify that the project work entitled **“AI-Powered Emotion Detection for Virtual Healthcare Consultations”**, is a bonafide work of **MD SAMREEN BEGUM (HT.No: 23U61A0544)** submitted in partial fulfillment of the requirement for the award of **Bachelor of Technology in Computer Science and Engineering** during the academic year 2024-25. This is further certified that the work done under my guidance, and the results of this work have not been submitted elsewhere for the award of any other degree or diploma.

### Internal Guide Head of the Department

**Mrs. G Pavani Mrs. Noore Ilahi**

**Assistant Professor Assistant Professor**

# DECLARATION

I hereby declare that the project work entitled **AI-Powered Emotion Detection for Virtual Healthcare Consultations**, submitted to **Department of Computer Science and Engineering, Global Institute of Engineering & Technology, Moinabad,** affiliated to **JNTUH, Hyderabad** in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is the work done by me and has not been submitted elsewhere for the award of any degree or diploma.

**MD SAMREEN BEGUM (23U61A0544)**

## ACKNOWLEDGEMENT

I am thankful to my guide **Mrs. G Pavani**, Assistant Professor of CSE Department for her valuable guidance for successful completion of this project.

I express my sincere thanks to **Mrs. G Pavani**, Project Coordinator for giving me an opportunity to undertake the project **“AI-Powered Emotion Detection for Virtual Healthcare Consultations”** and for enlightening me on various aspects of my project work and assistance in the evaluation of material and facts. She not only encouraged me to take up this topic but also given her valuable guidance in assessing facts and arriving at conclusions.

I am also most obliged and grateful to **Mrs. Noore Ilahi**, Assistant Professor and Head, Department of CSE for giving me guidance in completing this project successfully.

I express my heart-felt gratitude to our Vice-Principal **Prof. Dr. G Ahmed Zeeshan**, Co- Ordinator Internal Quality Assurance Cell (IQAC) for his constant guidance, cooperation, motivation and support which have always kept me going ahead. I owe a lot of gratitude to him for always being there for me.

I also most obliged and grateful to our Principal **Dr. P. Raja Rao** for giving me guidance in completing this project successfully.

I also thank my parents for their constant encourage and support without which the project would have not come to an end.

Last but not the least, I would also like to thank all my classmates who have extended their cooperation during our project work.

**MD SAMREEN BEGUM (23U61A0544)**

iv

## VISION

The vision of the department is to produce professional computer science engineers who can meet the expectations of the globe and contribute to the advancement of engineering and technology which involves creativity and innovations by providing an excellent learning environment with the best quality facilities.

## MISSION

1. To provide the students with a practical and qualitative education in a modern technical environment that will help to improve their abilities and skills in solving programming problems effectively with different ideas and knowledge.
2. To infuse the scientific temper in the students towards the research and development in Computer Science and Engineering trends.
3. To mould the graduates to assume leadership roles by possessing good communication skills, an appreciation for their social and ethical responsibility in a global setting, and the ability to work effectively as team members.

## PROGRAMME EDUCATIONAL OBJECTIVES

**PEO1:** To provide graduates with a good foundation in mathematics, sciences and engineering fundamentals required to solve engineering problems that will facilitate them to find employment in MNC’s and / or to pursue post graduate studies with an appreciation for lifelong learning.

**PEO2:** To provide graduates with analytical and problem-solving skills to design algorithms, other hardware / software systems, and inculcate professional ethics, inter-personal skills to work in a multi-cultural team.

**PEO3:** To facilitate graduates to get familiarized with the art software / hardware tools, imbibing creativity and innovation that would enable them to develop cutting edge technologies of multi- disciplinary nature for societal development.

## PROGRAMME OUTCOMES

**PO1:** Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

**PO2:** Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural science and engineering sciences.

**PO3:** Design/development of solutions: design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal and environmental considerations.

**PO4:** Conduct investigations of complex problems: use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5:** Modern tool usage: create, select and apply appropriate techniques, resources and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6:** The engineer and society: apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7:** Environment sustainability: understand the impact of the professional engineering solutions in the societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8:** Ethics: apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9:** Individual and team work: function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10:** Communication: communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11:** Project management and finance: demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12:** Lifelong learning: recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broader context of technological change.

## PROGRAMME SPECIFIC OUTCOMES

**PSO1:** An Ability to Apply the fundamentals of mathematics, Computer Science and Engineering Knowledge to analyze and develop computer programs in the areas related to Algorithms, System Software, Web Designing, Networking and Data mining for efficient Design of computer-based system to deal with Real time Problems.

**PSO2:** Ability to implement the Professional Engineering solutions for the betterment of Society, and able to communicate with professional Ethics effectively.

**ABSTRACT**

The article presents some models and strategies of consumer behavior in electronic commerce systems. An analysis of e-commerce recommender systems was also carried out, including context-aware recommender systems on the example of the analysis of regions of Ukraine and weather conditions, etc. The core of the novelty of the research is primarily related to hybrid behavioral models, personalization, contextual recommendations, especially integrating contextual information, such as location, time of day, or user intent, evaluation metrics, etc. The impact of machine learning and artificial intelligence on e-commerce recommender systems and e-commerce systems in general is also explored. The article also presents the microservice architecture of an electronic store with recommender system API. The architecture includes several key components, such as client-side APIs for managing orders, products and carts, services for processing customer requests and interacting with databases, and data processing technology using artificial intelligence.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **INDEX** | | | | |
| **CONTENTS** | | | | **PAGE.NO** |
| **COVER OR TITLE PAGE** | | | | i |
| **CERTIFICATE** | | | | ii |
| **DECLARATION** | | | | iii |
| **ACKNOWLEDGEMENT** | | | | iv |
| **ABSTRACT** | | | | 1 |
| **INDEX** | | | | 2 |
| **LIST OF FIGURES** | | | | 4 |
| **CHAPTERS** | | | | **PAGE.NO** |
| 1 | **INTRODUCTION** | | | 5 |
| 1.1 | INTRODUCTION | | 6 |
| 1.2 | EXISTING SYSTEM | | 7 |
| 1.3 | PROPOSED SYSTEM | | 7 |
|  | 1.4 | OVERVIEW OF PROJECT | | 8 |
| 2 | **LITERATURE SURVEY** | | | 9 |
| 2.1 | LITERATURE SURVEY | | 10 |
| 2.2 | FEATURES OF THE LANGUAGE USED | | 11 |
| 3 | **SYSTEM ANALYSIS** | | | 18 |
| 3.1 | SYSTEM REQUIREMENTS SPECIFICATIONS | | 19 |
| 3.1.1 | HARDWARE REQUIREMENTS | 19 |
| 3.1.2 | SOFTWARE REQUIREMENTS | 19 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | **SYSTEM DESIGN** | | | 21 |
| 4.1 | SYSTEM ARCHITECTURE | | 22 |
| 4.2 | DATA FLOW DIAGRAM | | 23 |
| 4.3 | UML DIAGRAMS | | 24 |
| 5 | **SYSTEM IMPLEMENTATION** | | | 30 |
| 5.1 | MODULES | | 31 |
| 5.2 | SOURCE CODE | | 32 |
| 6 | **SYSTEM TESTING** | | | 39 |
| 6.1 | SYSTEM TESTING | | 43 |
| 6.2 | TYPES OF TESTS | | 43 |
| 6.2.1 | UNIT TESTING | 43 |
| 6.2.2 | Integration testing | 43 |
| 6.2.3 | Functional test | 43 |
| 6.3 | TEST CASES | | 44 |
| 6.3.1 | Unit Testing | 44 |
| 6.3.2 | Integration Testing | 44 |
| 6.3.3 | Acceptance Testing | 44 |
| 7 | **RESULT** | | | 46-48 |
| 8 | **CONCLUSION** | | | 50 |
| **FUTURE SCOPE** | | |
| **REFERENCES** | | | | 51 |



**CHAPTER – 1 INTRODUCTION**

# INTRODUCTION

# The widespread popularity of the Internet and the implementation of web technologies worldwide have led to an increasing number of people abandoning traditional shopping methods and opting for online payments and purchases. As globalization, informatization, and computerization continue to advance, commercial transactions are increasingly taking place on specialized web pages such as online stores, e-commerce platforms, online showcases, online auctions, online trading platforms, etc. All of these are participants in e-commerce ecosystems. An e-commerce ecosystem can be understood as an area of activity where buyers and producers play complementary roles. Components of an e-commerce ecosystem typically include: customers, sellers, online marketplace, online payment system, logistics, marketing, data analysis, etc. Let's explore the essence of consumer behavior models and modern approaches in using recommender systems of e-commerce platforms. Consumer behavior models play a crucial role in modern e-commerce systems. These models help companies understand how consumers make decisions and interact with online platforms, enabling them to adapt their products, services, and marketing strategies according to consumer needs. One of the most widely used consumer behavior models is the buyer decision-making process model,

# which consists of five stages: 1) problem recognition, 2) information search, 3) evaluation of alternatives, 4) purchase decision, and 5) post- purchase evaluation and follow-up.

# EXISTING SYSTEM

Existing multi-objective recommendation systems in e-commerce are designed to balance competing goals to enhance user experience while aligning with business objectives. Unlike single-objective systems, which primarily focus on relevance (e.g., recommending items similar to past purchases), MORS incorporate additional dimensions such as:

* **Relevance**: Ensuring recommendations align with user preferences.
* **Diversity**: Offering varied product options to prevent repetitive suggestions.
* **Novelty**: Introducing new or less-familiar items to encourage discovery.
* **Serendipity**: Suggesting unexpected but potentially appealing items.
* **Business Metrics**: Promoting high-margin products, cross-selling, or managing inventory.

# PROPOSED SYSTEM

The proposed multi-objective recommendation system (MORS) for e-commerce platforms enhances user experience by delivering personalized product suggestions that optimize relevance, diversity, novelty, and business goals. Leveraging user browsing (e.g., product views, search queries) and purchasing behavior (e.g., transaction history, cart additions), the system employs a hybrid approach combining collaborative filtering, content-based filtering, and contextual modeling. Advanced multi-objective optimization techniques, such as Pareto optimization and contextual bandits, dynamically balance trade-offs, while privacy-preserving methods like differential privacy ensure compliance with regulations like GDPR. The modular, scalable architecture supports real-time adaptability, robust cold-start handling, and explainable recommendations, driving user engagement, conversion rates, and business outcomes like inventory management and revenue growth.

# OVERVIEW OF PROJECT

Overall, models of consumer behavior are essential for modern e-commerce systems to optimize their offerings, user experiences, and sales conversions by understanding how consumers think, feel, and behave when making purchasing. The specificity of e-commerce means that traditional models of consumer behavior in a regular store cannot be used when buying online. This is primarily due to the absence of physical attributes of the store, such as the cleanliness of the goods and the trading floor, convenience of entering and store location, and emotional and psychological factors that affect subjective consumer opinions. Additionally, trust factors related to e-stores, products, and transactions are of utmost importance in e commerce. As a result, consumer behavior models have undergone significant changes. The emergence and spread of the Internet cannot be viewed in isolation, and the models discussed below are a synthesis of offline and online behavior.



**CHAPTER – 2 LITERATURE SURVEY**

## LITERATURE SURVEY

### Koren et al. (2009)

### Koren et al. (2009) introduced matrix factorization techniques, such as Singular Value Decomposition (SVD), for collaborative filtering in recommendation systems. Their work, published in *IEEE Computer*, laid the foundation for user-item preference modeling, widely used in e-commerce platforms like Amazon for predicting relevant products based on purchase history.

### Lops et al. (2011)

Lops et al. (2011) reviewed content-based recommendation systems in *Recommender Systems Handbook*, emphasizing the use of product metadata (e.g., category, description) for recommendations. This approach is effective for cold-start scenarios in e-commerce, where new items lack interaction data.

.

### Burke (2002)

Burke (2002) explored hybrid recommendation systems in *User Modeling and User-Adapted Interaction*, combining CF and CBF to improve robustness. Hybrid models are prevalent in e-commerce for integrating browsing (e.g., product views) and purchasing data.

### Adomavicius and Kwon (2012)

Adomavicius and Kwon (2012) introduced multi-objective optimization for recommendations in *ACM Transactions on Information Systems*, focusing on balancing accuracy and diversity. They proposed re-ranking algorithms to diversify recommendations, reducing redundancy in e-commerce product suggestions

.

# FEATURES OF THE LANGUAGE USED

## PYTHON

* + - **Python** Is an Object-Oriented, High-Level Language, Interpreted, Dynamic and Multipurpose Programming Language.
    - Python Is Easy to Learn Yet Powerful and Versatile Scripting Language Which Makes It Attractive for Application Development.
    - Python's Syntax and Dynamic Typing with Its Interpreted Nature, Make It an Ideal Language for Scripting and Rapid Application Development in Many Areas.
    - Python Supports Multiple Programming Pattern, Including Object Oriented Programming, Imperative and Functional Programming or Procedural Styles.
    - Python Is Not Intended to Work on Special Area Such as Web Programming. That Is Why It Is Known as Multipurpose Because It Can Be Used with Web, Enterprise, 3d Cad Etc.

### Python Features

#### Easy To Use:

Python Is Easy to Very Easy to Use and High-Level Language. Thus, It Is Programmer-Friendly Language.

#### Expressive Language:

Python Language Is More Expressive. The Sense of Expressive Is the Code Is Easily Understandable.

#### Interpreted Language:

Python Is an Interpreted Language I.E. Interpreter Executes the Code Line by Line at A Time. This Makes Debugging Easy and Thus Suitable for Beginners.

#### Cross-Platform Language:

Python Can Run Equally on Different Platforms Such as Windows, Linux, Unix, Macintosh Etc. Thus, Python Is a Portable Language.

#### Free And Open Source:

Python Language Is Freely Available (www.Python. Org). The Source-Code Is Also Available. Therefore, It Is Open Source.

#### Object-Oriented Language:

Python Supports Object Oriented Language. Concept Of Classes And Objects Comes Into Existence.

#### Extensible:

It Implies That Other Languages Such As C/C++ Can Be Used To Compile The Code And Thus It Can Be Used Further In Your Python Code.

#### Large Standard Library:

Python Has A Large And Broad Library.

#### Gui Programming:

Graphical User Interfaces Can Be Developed Using Python.

#### Integrated:

It Can Be Easily Integrated With Languages Like C, C++, Java Etc.

### Python History

* + Python Laid Its Foundation In The Late 1980s.
  + The Implementation Of Python Was Started In The December 1989 By **Guido Van Rossum** At Cwi In Netherland.
  + Abc Programming Language Is Said To Be The Predecessor Of Python Language Which Was Capable Of Exception Handling And Interfacing With Amoeba Operating System.
  + Python Is Influenced By Programming Languages Like:
    - Abc Language.
    - Modula-3

### What Is Computer Vision?

Computer Vision Is The Broad Parent Name For Any Computations Involving Visual Content. That Means Images, Videos, Icons, And Anything Else With Pixels Involved. But Within This Parent Idea, There Are A Few Specific Tasks That Are Core Building Blocks:

* In **Object Classification**, You Train A Model On A Dataset Of Specific Objects, And

The Model Classifies New Objects As Belonging To One Or More Of Your Training Categories.

* For **Object Identification**, Your Model Will Recognize A Specific Instance Of An Object – For Example, Parsing Two Faces In An Image And Tagging One As Tom

Cruise And One As Katie Holmes.

A Classical Application Of Computer Vision Is Handwriting Recognition For Digitizing Handwritten Content (We’ll Explore More Use Cases Below). Outside Of Just Recognition, Other Methods Of Analysis Include:

* Video **Motion Analysis** Uses Computer Vision To Estimate The Velocity Of Objects In A Video, Or The Camera Itself.
* In **Image Segmentation**, Algorithms Partition Images Into Multiple Sets Of Views.
* **Scene Reconstruction** Creates A 3d Model Of A Scene Inputted Through Images Or Video (Check Out [Selva](https://www.selva3d.com/)).
* In **Image Restoration**, Noise Such As Blurring Is Removed From Photos Using Machine Learning Based Filters.

Any Other Application That Involves Understanding Pixels Through Software Can Safely Be Labeled As Computer Vision.

## ABOUT JAVA

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

Java is a programmer’s language. Java is cohesive and consistent.

Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

Finally, Java is to Internet programming where C was to system programming.

## IMPORTANCE OF JAVA TO THE INTERNET

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer.

They are: Passive information and Dynamic active programs. The Dynamic, Self- executing programs cause serious problems in the areas of Security and probability. But Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

## JAVA CAN BE USED TO CREATE TWO TYPES OF PROGRAMS

**Applications and Applets:** An application is a program that runs on our computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important.

An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

## FEATURES OF JAVA SECURITY

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

### Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

### Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine

## JAVASCRIPT

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then updates the browser’s display accordingly

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client-side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

Here are a few things we can do with JavaScript:

* Validate the contents of a form and make calculations.
* Add scrolling or changing messages to the Browser’s status line.
* Detect the browser in use and display different content for different browsers.
* Detect installed plug-ins and notify the user if a plug-in is required. We can do much more with JavaScript, including creating entire application. **JAVASCRIPT VS JAVA**
* JavaScript and Java are entirely different languages. A few of the most glaring

differences are: Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.

* While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact, they can be used together to combine their advantages.

## ADVANTAGES

* JavaScript can be used for Sever-side and Client-side scripting.
* It is More flexible than VBScript.
* JavaScript is the default scripting languages at Client-side since all the browsers supports it.

### Hyper Text Markup Language (HTML)

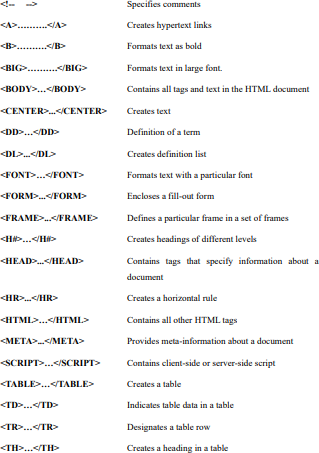
Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

**Basic HTML Tags:**

****



**CHAPTER – 3 SYSTEM ANALYSIS**

# SYSTEM REQUIREMENTS SPECIFICATIONS

* + 1. **HARDWARE REQUIREMENTS**

|  |  |
| --- | --- |
| **PROCESSOR** | Intel Core i5 or AMD Ryzen 5 |
| **HARD DISK** | 50 GB |
| **MONITOR** | 15 VGA Colour. |
| **MOUSE** | Optical USB Mouse |
| **RAM** | 8 GB and above |

* + 1. **SOFTWARE REQUIREMENTS**

|  |  |
| --- | --- |
| **OPERATING SYSTEM** | Windows 10/11 |
| **CODING LANGUAGE** | PYTHON, JS, HTML, CSS |

* 1. **Materials and Methods**

A qualitative research approach was employed in the study, which involved the analysis and synthesis of data and a systematic approach. The results were then used to draw conclusions about the significance of identified factors for the successful functioning of e-commerce recommender systems and how consumer behavior models can be utilized to create effective digital marketing strategies and personalized customer strategies. Overall, the materials and methods used in this research aimed to provide a comprehensive understanding of key factors contributing to the success of e-commerce systems, including descriptions of consumer behavior models and e-commerce recommender systems

## Models of Customer Behavior

There are many models that take into account the motives for product selection, the level of awareness and independence in choice, the level of consumer satisfaction with the product, and the 212 direction and possibility of influencing consumer choice through marketing and advertising stimuli. There are also other well-known consumer behavior models. The rational model corresponds to the concept of an “economic man”, rationalizing their actions on the path to personal gain. The irrational consumption model is based on behaviorism with the principle of “stimulus-response”: if the emotional attractiveness of a product is presented to the consumer for a sufficient amount of time in a comprehensible form, they will indeed begin to experience positive emotions from its acquisition.

## SOCIAL FEASIBILITY

The aspect of study is social feasibility hinges on addressing ethical concerns, such as ensuring transparency through explainable recommendations (e.g., “Recommended based on your recent views”), which builds user trust, as seen in potential extensions of app.py. Privacy is a critical factor; the system must comply with regulations like GDPR by securely handling user data (e.g., ratings in clean\_data.csv) and offering control over personalization settings. Inclusivity is also vital—avoiding bias in recommendations (e.g., over-recommending popular items) ensures diverse user groups, including those with varied demographics or new users, receive fair suggestions.

.



**CHAPTER – 4 SYSTEM DESIGN**

# SYSTEM ARCHITECTURE

Recommendation Engine

Browsing & Purchase Data

User Interactions

Data Collection

Data Preprocessing

Recommendation Generator

Personalized Suggestions

**FIGURE 4.1 SYSTEM ARCHITECTURE**

## DATA FLOW DIAGRAM

Also known as DFD, Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

# UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object- oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

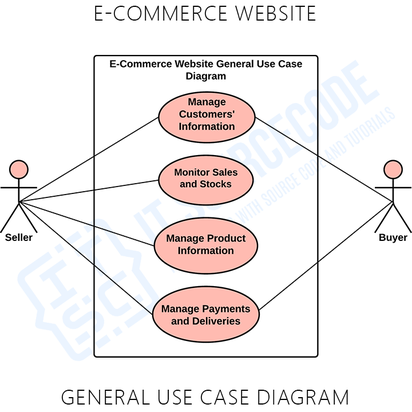
# GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

## USE CASE DIAGRAM

* + - * UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.
      * UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.
      * OMG is continuously putting effort to make a truly industry standard.
      * UML stands for **U**nified **M**odeling **L**anguage.
      * UML is a pictorial language used to make software blue prints



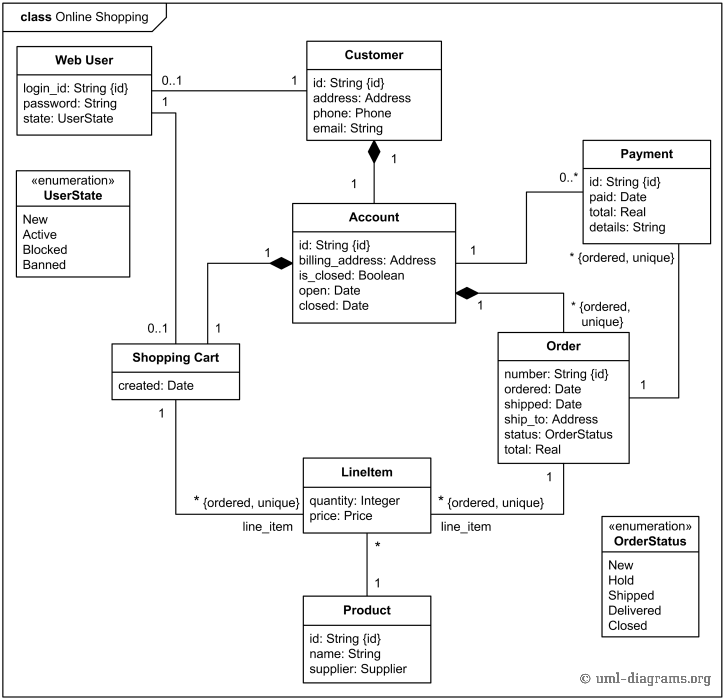
**FIGURE 4.3.1 USE CASE**

## CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

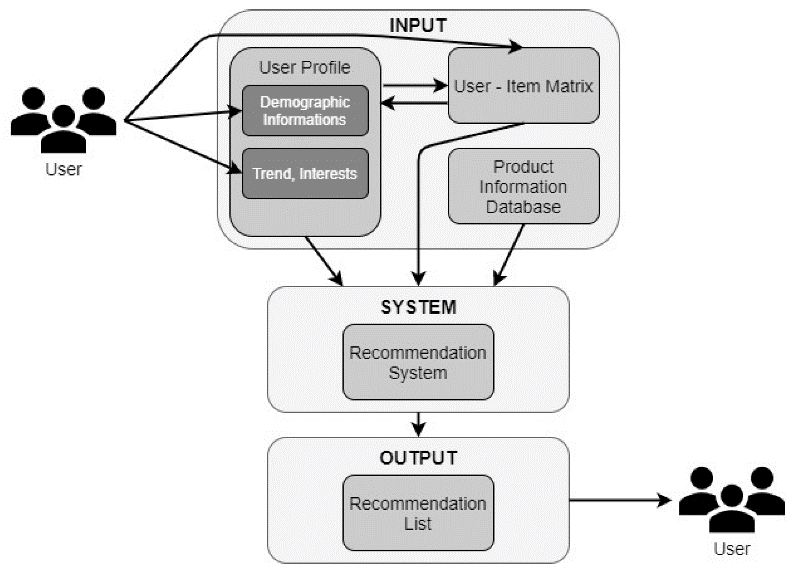
* + - * The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
      * The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
      * The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

****

**FIGURE 4.3.2 CLASS DIAGRAM**

## SEQUENCE DIAGRAM

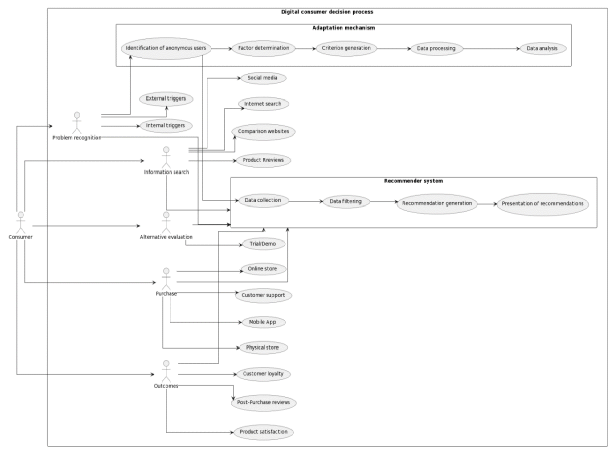
Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure.



**FIGURE 4.3.3 SEQUENCE DIAGRAM**

## ACTIVITY DIAGRAM

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency.In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

****

## INPUT DESIGN AND OUTPUT DESIGN INPUT DESIGN

The input design is system requires three primary types of input data: user interaction data, product metadata, and contextual data. User interaction data includes browsing history (e.g., page views, time spent on product pages, search queries), purchase history (e.g., items bought, purchase frequency, recency), and explicit feedback (e.g., ratings, reviews, likes), which collectively provide insights into user preferences and behavior patterns, as partially implemented in the clean\_data.csv file from the provided app.py code, which contains user IDs, product IDs, and ratings. Product metadata encompasses detailed attributes of items, such as product IDs, names, categories, prices, images (e.g., ImageURL in app.py), inventory status, and additional features like brand or popularity scores (e.g., trending\_products.csv in app.py), enabling the system to characterize items for content-based filtering and business-driven objectives. Contextual data includes situational factors like time of day, user location, device type (e.g., mobile vs. desktop), or seasonal trends, which allow for dynamic, context-aware recommendations, though not explicitly used in app.py.

The input design prioritizes minimal manual intervention, supporting seamless user experience and broad compatibility across various devices and telehealth platforms.

## OUTPUT DESIGN

The output design focuses on delivering The primary output is a ranked list of recommended products, each accompanied by essential metadata including product IDs, names, categories, prices, and image URLs, as exemplified in the recommend.html template of the provided app.py code, which displays a selected product and its top similar items. This list is generated to optimize relevance (matching user preferences based on browsing and purchase history), diversity (ensuring variety in product categories to encourage exploration), and business objectives (e.g., promoting trending or high-margin items, as seen with trending\_products in app.py). Additional output components may include confidence scores to indicate the strength of each recommendation, enhancing transparency, and recommendation explanations (e.g., “Recommended because you viewed similar electronics”) to build user trust and engagement such as:

* **Ranked Product List**: A prioritized list of recommended products with metadata (product ID, name, category, price, image URL), as rendered in recommend.html from app.py, balancing relevance, diversity, and business goals (e.g., promoting trending items).
* **Confidence Scores**: Numerical indicators of recommendation strength to enhance transparency.
* **Recommendation Explanations**: User-friendly text (e.g., “Based on your recent views”) to justify suggestions and build trust.



**CHAPTER – 5 SYSTEM**

**IMPLEMENTATION**

# MODULES

## Data Collection Module

This module is responsible for gathering user interaction data and product metadata to fuel recommendations. Collects and aggregates raw data on user interactions (browsing, purchases, ratings) and product metadata (e.g., ID, name, category, price, images) to support personalized recommendations.

**DATS PREPROCESSING MODULE**

This module processes the Cleans, transforms, and structures raw data into formats suitable for analysis, handling missing values, duplicates, and data encoding.

**FEATURE ENGINEERING MODULE**

In this module, Extracts and constructs features to represent users (e.g., preference profiles) and products (e.g., category, price) for use in recommendation algorithms.

**RECOMMENDATION GENERATION** **MODULE**

This core module applies Applies algorithms (collaborative filtering, content-based filtering, hybrid) to generate personalized product suggestions based on user behavior.

**MULTI-OBJECTIVE OPTIMIZATION MODULE**

The Balances multiple objectives (relevance, diversity, business goals) by computing weighted scores or applying optimization techniques for recommendation ranking.

**USER INTERFACE** **MODULE**

This module Presents recommendations and product information through a user-friendly interface, supporting web and mobile platforms.

SOURCE CODE

**INDEX.HTML**

{% extends "base.html" %}

{% block title %}Home{% endblock %}

{% block content %}

  <!-- Filter Form -->

  <form id="preference-form" class="preference-form">

    <label>

      Category:

      <select name="category" id="category-select">

        <option value="">-- Select Category --</option>

        <option value="beauty">Beauty</option>

        <option value="hair">Hair</option>

        <option value="makeup">Makeup</option>

        <option value="nail">Nail</option>

        <!-- Add more categories as needed -->

      </select>

    </label>

    <label>

      Brand:

      <select name="brand" id="brand-select">

        <option value="">-- Select Brand --</option>

        <option value="OPI">OPI</option>

        <option value="Clairol">Clairol</option>

        <option value="Kokie">Kokie</option>

        <!-- Add more brands as needed -->

      </select>

    </label>

    <label>

      Minimum Rating:

      <select name="rating" id="rating-select">

        <option value="">-- Any --</option>

        <option value="5">5</option>

        <option value="4">4+</option>

        <option value="3">3+</option>

        <option value="2">2+</option>

        <option value="1">1+</option>

      </select>

    </label>

    <button type="submit">Filter</button>

  </form>

  <h2>Trending Items</h2>

  <div class="grid">

    {% for \_, row in trending.iterrows() %}

      <a class="tile"

         href="{{ url\_for('recommend', prodid=row.ProdID) }}"

         data-category="{{ row.Category|lower }}"

         data-brand="{{ row.Brand|lower }}"

         data-rating="{{ row.Rating }}">

        <img src="{{ row.ImageURL }}" alt="{{ row.Name }}">

        <p>{{ row.Name }}</p>

      </a>

    {% endfor %}

  </div>

  <h2>All Products</h2>

  <div class="grid">

    {% for \_, row in products.iterrows() %}

      <a class="tile"

         href="{{ url\_for('recommend', prodid=row.ProdID) }}"

         data-category="{{ row.Category|lower }}"

         data-brand="{{ row.Brand|lower }}"

         data-rating="{{ row.Rating }}">

        <img src="{{ row.ImageURL }}" alt="{{ row.Name }}">

        <p>{{ row.Name }}</p>

      </a>

    {% endfor %}

  </div>

{% endblock %}

**STYLES.CSS**

body {

font-family: 'Segoe UI', Arial, sans-serif;

margin: 0;

padding: 0;

background: #f9f9fb;

}

header {

background: #333;

color: #fff;

padding: 1rem;

}

header a {

color: #fff;

text-decoration: none;

font-size: 2rem;

}

main {

padding: 2rem 1rem 1rem 1rem;

max-width: 1200px;

margin: 0 auto;

}

.preference-form {

display: flex;

gap: 1.5rem;

margin-bottom: 2rem;

align-items: flex-end;

background: #f7f7fa;

padding: 1rem 1.5rem;

border-radius: 8px;

box-shadow: 0 2px 8px #eee;

}

.preference-form label {

display: flex;

flex-direction: column;

font-size: 1rem;

color: #333;

}

.preference-form select {

margin-top: 0.25rem;

padding: 0.4rem;

border-radius: 4px;

border: 1px solid #bbb;

font-size: 1rem;

background: #fff;

}

.preference-form button {

padding: 0.5rem 1.2rem;

background: #333;

color: #fff;

border: none;

border-radius: 4px;

font-size: 1rem;

cursor: pointer;

transition: background 0.2s;

margin-top: 1.2rem;

}

.preference-form button:hover {

background: #444;

}

h2, h3 {

margin-top: 2rem;

color: #2c2c2c;

}

.grid {

display: grid;

grid-template-columns: repeat(auto-fill, minmax(160px, 1fr));

gap: 1.2rem;

margin-bottom: 2rem;

}

.tile {

text-align: center;

text-decoration: none;

color: inherit;

background: #fff;

border-radius: 8px;

box-shadow: 0 1px 6px #e0e0e0;

padding: 0.8rem 0.2rem 0.8rem 0.2rem;

transition: box-shadow 0.2s, transform 0.1s;

display: flex;

flex-direction: column;

align-items: center;

}

.tile img {

width: 100%;

height: 120px;

object-fit: cover;

border-radius: 6px;

border: 1px solid #ddd;

margin-bottom: 0.5rem;

transition: filter 0.2s;

}

.tile:hover {

box-shadow: 0 4px 16px #d1d1d1;

transform: translateY(-2px) scale(1.03);

}

.tile img:hover {

filter: brightness(0.88);

}

.large {

max-width: 300px;

display: block;

margin-bottom: 1rem;

border-radius: 10px;

border: 1px solid #ccc;

box-shadow: 0 2px 10px #eee;

}

.product-meta {

display: flex;

gap: 2rem;

margin-bottom: 2rem;

font-size: 1.1rem;

color: #555;

}

@media (max-width: 700px) {

.grid {

grid-template-columns: repeat(auto-fill, minmax(120px, 1fr));

gap: 0.8rem;

}

.preference-form {

flex-direction: column;

gap: 0.6rem;

padding: 1rem;

}

.product-meta {

flex-direction: column;

gap: 0.5rem;

  }

}

**SCRIPT.JS**

// static/js/scripts.js

document.addEventListener('DOMContentLoaded', function () {

  // Filtering logic for the preference form

  const form = document.getElementById('preference-form');

  if (form) {

    form.addEventListener('submit', function (e) {

      e.preventDefault();

      const category = document.getElementById('category-select').value.toLowerCase();

      const brand = document.getElementById('brand-select').value.toLowerCase();

      const rating = parseFloat(document.getElementById('rating-select').value);

      // Select all product tiles in the "All Products" section (the last .grid)

      const allProductGrid = document.querySelectorAll('.grid:last-of-type .tile');

      allProductGrid.forEach(tile => {

        const tileCategory = (tile.getAttribute('data-category') || '').toLowerCase();

        const tileBrand = (tile.getAttribute('data-brand') || '').toLowerCase();

        const tileRating = parseFloat(tile.getAttribute('data-rating')) || 0;

        let show = true;

        if (category && !tileCategory.includes(category)) show = false;

        if (brand && !tileBrand.includes(brand)) show = false;

        if (rating && tileRating < rating) show = false;

        tile.style.display = show ? '' : 'none';

      });

    });

  }

  // Optional: visually highlight trending products

  if (window.location.pathname === '/') {

    document.querySelectorAll('.grid:first-of-type .tile').forEach(tile => {

      tile.style.boxShadow = '0 0 8px 2px #ffd70077';

    });

  }

  // Image hover effect for product tiles

  document.querySelectorAll('.tile img').forEach(img => {

    img.addEventListener('mouseover', () => {

      img.style.filter = 'brightness(0.88)';

      img.style.transition = 'filter 0.2s';

    });

    img.addEventListener('mouseout', () => {

      img.style.filter = '';

    });

  });

  // Click animation for product tiles

  document.querySelectorAll('.tile').forEach(tile => {

    tile.addEventListener('mousedown', () => {

      tile.style.transform = 'scale(0.97)';

    });

    tile.addEventListener('mouseup', () => {

      tile.style.transform = '';

    });

    tile.addEventListener('mouseleave', () => {

      tile.style.transform = '';

    });

  });

  // Lazy load images for better performance (if supported)

  if ('loading' in HTMLImageElement.prototype) {

    document.querySelectorAll('img').forEach(img => {

      img.setAttribute('loading', 'lazy');

    });

  }

});

**app.py**

from flask import Flask, render\_template, request, abort

import pandas as pd

import joblib

import numpy as np

from sklearn.decomposition import TruncatedSVD

app = Flask(\_\_name\_\_)

# --- Load data & models on startup ---

# Load main data

df = pd.read\_csv("clean\_data.csv")

# Load trending products and ensure 'ProdID' column exists

trending\_df = pd.read\_csv("trending\_products.csv")

trending\_df.rename(columns={trending\_df.columns[0]: 'ProdID'}, inplace=True)

trending = trending\_df['ProdID'].tolist()

# Load SVD model

svd = joblib.load("svd\_model.joblib")

# Build user-item matrix & prediction matrix

rating\_matrix = df.pivot\_table(

index='ID', columns='ProdID', values='Rating'

).fillna(0)

# Precompute full prediction matrix

full\_pred = svd.inverse\_transform(svd.transform(rating\_matrix.values))

pred\_df = pd.DataFrame(

full\_pred,

index=rating\_matrix.index,

columns=rating\_matrix.columns

)

# Helper: get top-N similar items for a given ProdID

def get\_item\_recs(prodid, n=5):

if prodid not in pred\_df.columns:

return [] # Return empty if product not found

item\_vec = pred\_df[prodid].values.reshape(1, -1)

all\_vecs = pred\_df.values.T # shape (items, users)

sims = np.dot(all\_vecs, item\_vec.T).flatten()

idxs = np.argsort(sims)[::-1]

prod\_ids = pred\_df.columns[idxs]

# skip the prodid itself

prod\_ids = [p for p in prod\_ids if p != prodid]

return prod\_ids[:n]

# --- Routes ---

@app.route("/")

def index():

# Show trending and all products

products = df.drop\_duplicates('ProdID')[['ProdID', 'Name', 'ImageURL']]

trending\_products = products[products['ProdID'].isin(trending)]

return render\_template(

"index.html",

products=products,

trending=trending\_products

)

@app.route("/recommend/<prodid>")

def recommend(prodid):

# Validate product ID

products\_unique = df.drop\_duplicates(subset='ProdID').set\_index('ProdID')

if prodid not in products\_unique.index:

abort(404, description="Product not found")

# 1. Get top 5 item-to-item recommendations

rec\_ids = get\_item\_recs(prodid, n=5)

# 2. Assemble the recommendation DataFrame

recs = products\_unique.loc[rec\_ids]

# 3. Fetch the selected product’s own info

product = products\_unique.loc[prodid]

# 4. Render the template

return render\_template(

"recommend.html",

product=product,

recommendations=recs.reset\_index()

)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

### Code Explanation:

1. **Imports and Initialization**

import os, numpy as np, librosa, soundfile, joblib

from flask import Flask, request, jsonify, render\_template from werkzeug.utils import secure\_filename

Flask handles web server and routing.

librosa & soundfile are used for audio processing. joblib loads the pre-trained emotion recognition model. numpy for numerical operations

### Configuration

UPLOAD\_FOLDER = 'uploads' ALLOWED\_EXTENSIONS = {'wav', 'mp3', 'ogg', 'flac'}

Specifies allowed audio formats and where uploaded files will be saved temporarily.

### Model Loading

MODEL\_PATH = os.path.join(os.path.dirname( file ), '..', 'model', 'mlp\_model.joblib') model = joblib.load(MODEL\_PATH)

Loads the pre-trained machine learning model that predicts emotions based on audio features. The model must have been trained earlier on labeled emotional speech datasets

### Emotion Mapping

emotions = {

'01': 'neutral', '02': 'calm', '03': 'happy', '04': 'sad',

'05': 'angry', '06': 'fearful', '07': 'disgust', '08': 'surprised'

}

Maps numeric emotion labels (as used during training) to human-readable form.

### Utility Functions

def allowed\_file(filename):

return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED\_EXTENSIONS

### Feature Extraction

def extract\_feature(file\_name, mfcc=True, chroma=True, mel=True):

Extracts MFCC, chroma, and mel spectrogram features from the audio—these are commonly used for speech emotion recognition.

These features capture:

MFCC: Timbre (tone quality) Chroma: Pitch class content

### Frontend Route

@app.route('/') def index():

return render\_template('index.html')

Loads the HTML interface (e.g., where a healthcare worker or user uploads audio

### Prediction Route

@app.route('/predict', methods=['POST']) def predict():

Accepts an audio file. Saves it temporarily. Extracts features.

Uses the model to predict the emotion. Returns the result as JSON.

Cleans up the temporary file after processing.

### Running the App

if name == ' main ': app.run(debug=True)

Starts the Flask server in debug model

# Algorithm:

### Step 1: Setup and Configuration

* 1. Import required libraries for audio processing, file handling, model loading, and web framework (Flask, librosa, numpy, joblib, etc.).
  2. Define allowed file extensions and the upload folder.
  3. Ensure the upload directory exists using os.makedirs.

### Step 2: Load Pre-trained Model

1. Construct the path to the pre-trained MLP model (mlp\_model.joblib).
2. Check if the model file exists.

### Step 3: Define Emotion Labels

Define a dictionary mapping emotion codes ('01', '02', etc.) to emotion names ('neutral', 'happy', etc.).

### Step 4: Audio File Validation

1. Create a function allowed\_file(filename) to validate the file extension.

### Step 5: Feature Extraction from Audio

Define the function extract\_feature(file\_name):

1. Load the audio file using soundfile.SoundFile()
2. Read the waveform data and sample rate.
3. Optionally compute:

MFCC (Mel Frequency Cepstral Coefficients): captures timbral features. Chroma features: represents the 12 different pitch classes.

Mel Spectrogram: represents perceived loudness.

1. Combine these features into a single feature vector.

### Step 6: Flask Routes

/ Route

Serves the HTML interface (index.html) for uploading the audio file.

/predict Route

Handles POST requests:

1. Validate that the request includes an audio file.
2. Save the uploaded file securely.
3. Extract audio features using extract\_feature.
4. Reshape features for model input.
5. Predict the emotion using model.predict().
6. Return the predicted emotion as JSON.

### Step 7: Start the Flask App

If the script is run directly (if name == ' main '), start the Flask development server



**CHAPTER – 6 SYSTEM TESTING**

# SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

# TYPES OF TESTS

## UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

### Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

# TEST CASES

### Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be

### Test objectives

* + - * All field entries must work properly.
      * Pages must be activated from the identified link.
      * The entry screen, messages and responses must not be delayed.

### Features to be tested

* + - * Verify that the entries are of the correct format
      * No duplicate entries should be allowed
      * All links should take the user to the correct page.

### Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

### Acceptance Testing

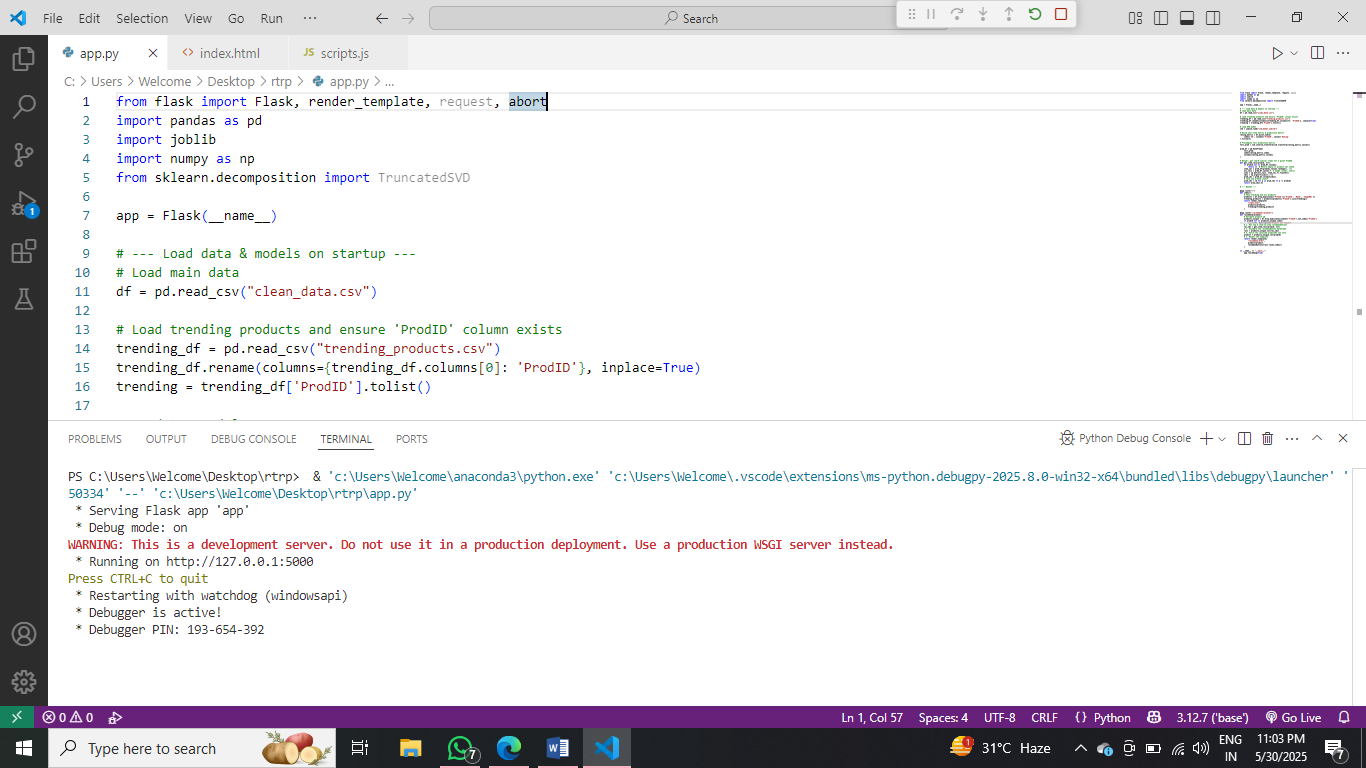
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user.

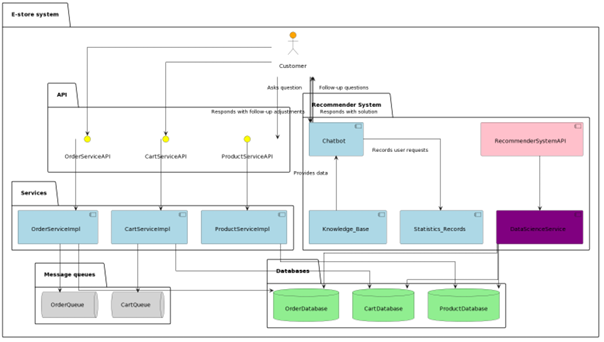
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.



**CHAPTER – 7 RESULT**

**CODE INITIALIZING AND LOADING THE MODEL**

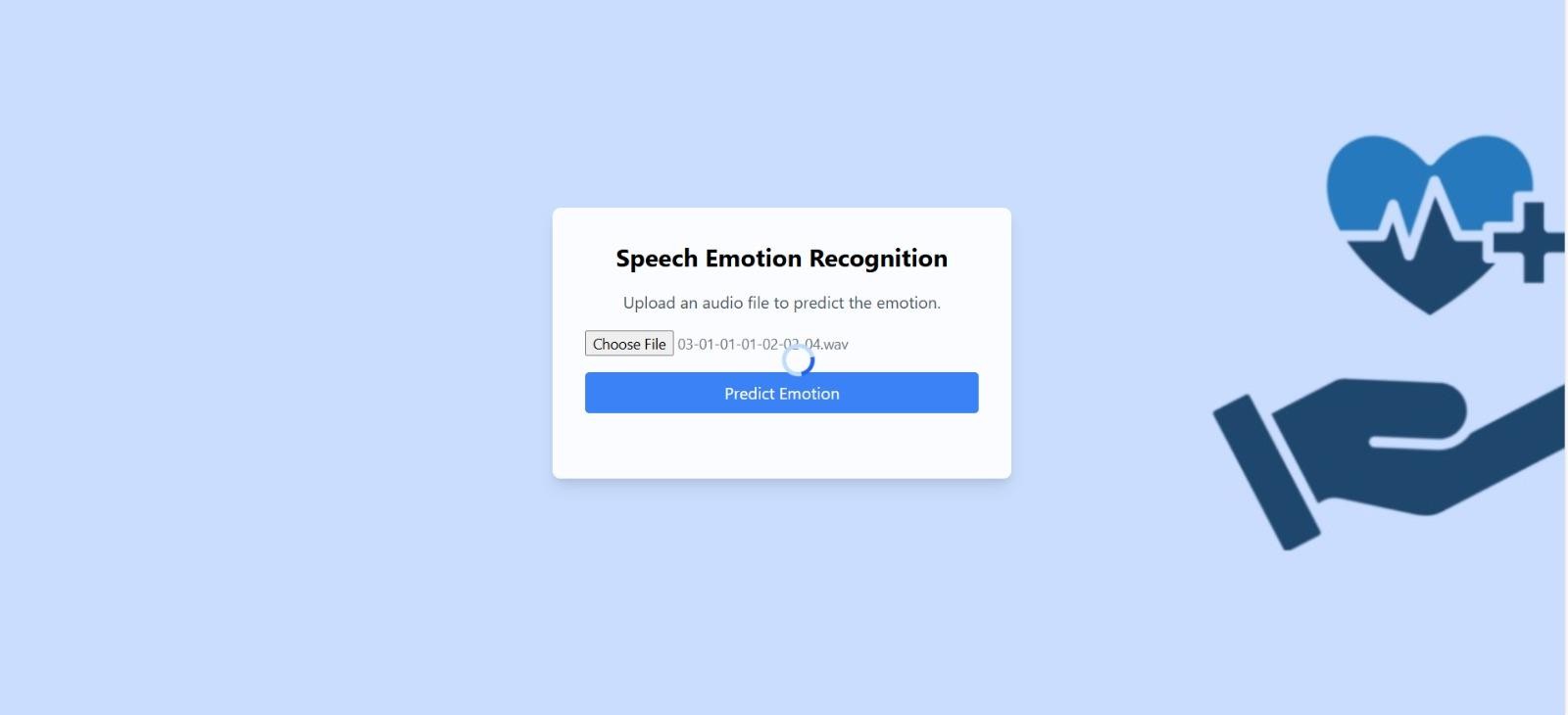
**FIGURE 7.1 CODE INITIALIZING AND LOADING THE MODEL**

**FIGURE 2**

**AUDIO FILE SELECTION FOR EMOTION PREDICTION**

**FIGURE 7.4 AUDIO FILE UPLOADED FOR EMOTION DETECTION**

**LOADING AUDIO FILE FOR EMOTION DETECTION**

****

**FIGURE 7.5 LOADING AUDIO FILE FOR EMOTION DETECTION**



**CHAPTER – 8 CONCLUSION**

# CONCLUSION

In light of the transformation into a digital society, the development of the field of electronic economy, and in particular, electronic business and electronic commerce, is receiving a new impetus. Therefore, consumer behavior models and recommender systems in e-commerce are of particular importance. In the field of recommender systems for e-commerce, several future trends and directions are emerging. These trends are as follows: personalization and contextualization, explainability and transparency, usage of hybrid approaches, context-aware recommendations, group and social recommendations, reinforcement learning, privacy and trust, multimodal recommendations, long-tail recommendations, online learning and real-time recommendations, etc. Real-time recommendation systems that continuously adapt to evolving user preferences and dynamics are gaining importance. Online learning techniques enable systems to update recommendations in real-time, considering the most recent user interactions and changes in preferences. These trends represent ongoing research and development efforts in the field of recommender systems for e-commerce, aiming to provide more accurate, personalized, and context-aware recommendations while addressing user concerns regarding transparency, privacy, and diversity. By understanding the behavior and preferences of your customers, using recommender systems for e-commerce companies, you can create better products, provide personalized experiences, and develop effective marketing strategies that contribute to the development of e-commerce in general.

# FUTURE SCOPE

For future advancements, the proposed project can be further modelled in terms of efficiency, accuracy, and usability. Additional to the emotions, the model can be extended to recognize feelings such as depression and mood changes. The future scope of a multi-objective recommendation system for e-commerce platforms includes:

1. **Advanced AI Integration**: Use deep learning (e.g., transformers, GNNs) and reinforcement learning for more accurate, adaptive recommendations.
2. **Real-Time Contextualization**: Incorporate real-time data (via streaming platforms like Kafka) and contextual signals (location, time, device) for dynamic personalization.
3. **Enhanced Multi-Objective Optimization**: Apply dynamic weight tuning and fairness-aware algorithms to balance relevance, diversity, and business goals.
4. **Scalability**: Deploy on cloud platforms with microservices and edge computing for low-latency, high-volume processing.

Therefore, in the future, there would emerge many applications .

# REFERENCES

[1] M. Wenzl, Understanding Online Consumer Behaviors for a Better Customer Journey. Available online: URL https://www.shipbob.com/blog/online-consumer-behavior/ (April 2021). 220

[2] J. Langer, S. Beckmann, How online shop owners can stimulate customers’ exploration of new products. Journal of Retailing and Consumer Services, 2017, 34, pp. 32-42.

[3] C. Liao, J. Chen, Enhancing consumer engagement in e-commerce websites: An integrated model of e-service quality, customer value, and switching costs. Journal of Retailing and Consumer Services, 2019, 50, pp. 23-32.

[4] E. Berghausen, C. Backhaus, When do consumers choose same-day versus next-day delivery in online shopping? An investigation of the underlying decision-making processes. Journal of Retailing and Consumer Services, 2020, 55, pp.102-109.

[5] T. Wagner, A. Benlian, T. Hess, The concept of digital affordances revisited: An exploratory study in the context of business-to-consumer electronic commerce. Journal of Business Research, 2019, 98, pp. 365-376.

[6] L. Otto, D. Schniederjans, How should e-commerce platforms design their recommendation systems to cope with multiple objectives? European Journal of Operational Research, 2017, 258(3), pp. 934 947.

[7] I. Fedorko, R. Bacik, B. Gavurova, Technology acceptance model in e-commerce segment. Management & Marketing, 13(4), pp. 1242-1256.

[8] Faradivah Dwi Azizah et al. Impulsive Buying Behavior: Implementation of IT on Technology Acceptance Model on E-Commerce Purchase Decisions. Golden Ratio of Marketing and Applied Psychology of Business, 2022, 2(1). doi:10.52970/grmapb.v2i1.173.